(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 March 2001 (08.03.2001)

PCT

(10) International Publication Number
WO 01/16750 A2

(51) International Patent Classification[7]: G06F 11/20

(21) International Application Number: PCT/US00/24329

(22) International Filing Date: 31 August 2000 (31.08.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

60/152,151	31 August 1999 (31.08.1999)	US
60/220,794	26 July 2000 (26.07.2000)	US
60/220,748	26 July 2000 (26.07.2000)	US

(63) Related by continuation (CON) or continuation-in-part (CIP) to earlier applications:

US	60/152,151 (CIP)
Filed on	31 August 1999 (31.08.1999)
US	60/220,974 (CIP)
Filed on	26 July 2000 (26.07.2000)
US	60/220,748 (CIP)
Filed on	26 July 2000 (26.07.2000)

(71) Applicant *(for all designated States except US)*: TIMES N SYSTEMS, INC. [US/US]; Building B, Suite P, 1908 Kramer Lane, Austin, TX 78758 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: WEST, Lynn, Parker [US/US]; 10201 Pantera Drive, Austin, TX 78759 (US). SCARDAMALIA, Ted [US/US]; 1910 Clearwater Dr., Round Rock, TX 78681 (US).

(74) Agent: BRUCKNER, John, J.; Fulbright & Jaworski LLP, 600 Congress Avenue, Suite 2400, Austin, TX 78701 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:

— *Without international search report and to be republished upon receipt of that report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 01/16750 A2

(54) Title: HIGH-AVAILABILITY, SHARED-MEMORY CLUSTER

(57) Abstract: Methods, systems and devices are described for a high-availability, shared-memory cluster. An apparatus, includes a shared memory unit; a first processing node coupled to the shared memory unit; and a second processing node coupled to the shared memory unit. The shared memory unit includes a range of addresses that are duplicated. The methods, systems and devices provide advantages because the speed and scalability of parallel processor systems is enhanced.

HIGH-AVAILABILITY, SHARED-MEMORY CLUSTER

BACKGROUND OF THE INVENTION

1. Field of the Invention

The invention relates generally to the field of multiprocessor computer systems. More particularly, the invention relates to computer systems that utilize a high-availability, shared-memory cluster.

2. Discussion of the Related Art

The clustering of workstations is a well-known art. In the most common cases, the clustering involves workstations that operate almost totally independently, utilizing the network only to share such services as a printer, license-limited applications, or shared files.

In more-closely-coupled environments, some software packages (such as NQS) allow a cluster of workstations to share work. In such cases the work arrives, typically as batch jobs, at an entry point to the cluster where it is queued and dispatched to the workstations on the basis of load.

In both of these cases, and all other known cases of clustering, the operating system and cluster subsystem are built around the concept of message-passing. The term message-passing means that a given workstation operates on some portion of a job until communications (to send or receive data, typically) with another workstation is necessary. Then, the first workstation prepares and communicates with the other workstation.

Another well-known art is that of clustering processors within a machine, usually called a Massively Parallel Processor or MPP, in which the techniques are essentially identical to those of clustered workstations. Usually, the bandwidth and latency of the interconnect network of an MPP are more highly optimized, but the system operation is the same.

In the general case, the passing of a message is an extremely expensive operation; expensive in the sense that many CPU cycles in the sender and receiver are consumed by the process of sending, receiving, bracketing, verifying, and routing the message, CPU cycles that are therefore not available

for other operations. A highly streamlined message-passing subsystem can typically require 10,000 to 20,000 CPU cycles or more.

There are specific cases wherein the passing of a message requires significantly less overhead. However, none of these specific cases is adaptable to a general-purpose computer system.

Message-passing parallel processor systems have been offered commercially for years but have failed to capture significant market share because of poor performance and difficulty of programming for typical parallel applications. Message-passing parallel processor systems do have some advantages. In particular, because they share no resources, message-passing parallel processor systems are easier to provide with high-availability features. What is needed is a better approach to parallel processor systems.

There are alternatives to the passing of messages for closely-coupled cluster work. One such alternative is the use of shared memory for inter-processor communication.

Shared-memory systems, have been much more successful at capturing market share than message-passing systems because of the dramatically superior performance of shared-memory systems, up to about four-processor systems. In Search of Clusters, Gregory F. Pfister 2nd ed. (January 1998) Prentice Hall Computer Books, ISBN: 0138997098 describes a computing system with multiple processing nodes in which each processing node is provided with private, local memory and also has access to a range of memory which is shared with other processing nodes. The disclosure of this publication in its entirety is hereby expressly incorporated herein by reference for the purpose of indicating the background of the invention and illustrating the state of the art.

However, providing high availability for traditional shared-memory systems has proved to be an elusive goal. The nature of these systems, which share all code and all data, including that data which controls the shared operating systems, is incompatible with the separation normally required for high availability. What is needed is an approach to shared-memory systems that improves availability.

Although the use of shared memory for inter-processor communication is a well-known art, prior to the teachings of U.S. Ser. No. 09/273,430, filed March 19, 1999, entitled Shared Memory Apparatus and Method for Multiprocessing Systems, the processors shared a single copy of the operating system. The problem with such systems is that they cannot be efficiently scaled beyond four to eight way systems except in unusual circumstances. All known cases of said unusual circumstances are such that the systems are not good price-performance systems for general-purpose computing.

The entire contents of U.S. Patent Applications 09/273,430, filed March 19, 1999 and PCT/US00/01262, filed January 18, 2000 are hereby expressly incorporated by reference herein for all purposes. U.S. Ser. No. 09/273,430, improved upon the concept of shared memory by teaching the concept which will herein be referred to as a tight cluster. The concept of a tight cluster is that of individual computers, each with its own CPU(s), memory, I/O, and operating system, but for which collection of computers there is a portion of memory which is shared by all the computers and via which they can exchange information. U.S. Ser. No. 09/273,430 describes a system in which each processing node is provided with its own private copy of an operating system and in which the connection to shared memory is via a standard bus. The advantage of a tight cluster in comparison to an SMP is "scalability" which means that a much larger number of computers can be attached together via a tight cluster than an SMP with little loss of processing efficiency.

What is needed are improvements to the concept of the tight cluster. What is also needed is an expansion of the concept of the tight cluster.

Another well-known art is the use of memory caches to improve performance. Caches provide such a significant performance boost that most modern computers use them. At the very top of the performance (and price) range all of memory is constructed using cache-memory technologies. However, this is such an expensive approach that few manufacturers use it. All manufacturers of personal computers (PCs) and workstations use caches except for the very low end of the PC business where caches are omitted for price reasons and performance is, therefore, poor.

Caches, however, present a problem for shared-memory computing systems; the problem of coherence. As a particular processor reads or writes a word of shared memory, that word and usually a number of surrounding words are transferred to that particular processor's cache memory transparently by cache-memory hardware. That word and the surrounding words (if any) are transferred into a portion of the particular processor's cache memory that is called a cache line or cache block.

If the transferred cache line is modified by the particular processor, the representation in the cache memory will become different from the value in shared memory. That cache line within that particular processor's cache memory is, at that point, called a "dirty" line. The particular processor with the dirty line, when accessing that memory address will see the new (modified) value. Other processors, accessing that memory address will see the old (unmodified) value in shared memory. This lack of coherence between such accesses will lead to incorrect results.

Modern computers, workstations, and PCs which provide for multiple processors and shared memory, therefore, also provide high-speed, transparent cache coherence hardware to assure that if a line in one cache changes and another processor subsequently accesses a value which is in that address range, the new values will be transferred back to memory or at least to the requesting processor. However, the expense of the additional hardware is significant and performance is degraded.

Caches can be maintained coherent by software provided that sufficient cache-management instructions are provided by the manufacturer. However, in many cases, an adequate arsenal of such instructions are not provided. Moreover, even in cases where the instruction set is adequate, the software overhead is so great that no examples of are known of commercially successful machines which use software-managed coherence.

Thus, the existing hardware and software cache coherency approaches are unsatisfactory. What is also needed, therefore, is a better approach to cache coherency.

Another well-known art is that of a "heartbeat" function. In a system involving multiple independent computers, a function can be provided such that each of said computers occasionally signals to at least a subset of the other processors an indication that status is operational. Failure to signal this heartbeat is a primary indication that the computer has failed, in either hardware or software. Should a companion processor fail to receive the heartbeat within a specified period of time following the previous received heartbeat signal, said companion processor will execute a verification routine. Should the results of the verification routine indicate computer failure, the system will enter checkpoint restart mode and will restart. The failed computer will be removed from the group upon restart and an operator message will be issued as part of the restart.

In symmetric multiprocessors (SMP's) such a heartbeat function is normally not applicable, as all the processors are using a single copy of the software, and software failure is the most common failure. Also, in event of hardware failure, the cache of the failed processor may contain dirty, required operating system status, so that recovery is often impossible. Since there is no way to determine from other processors whether recovery is possible, there are no known examples of SMP systems which attempt to recover from processor or memory failures.

While the heartbeat functionality can be provided in the context of a message passing system, such systems have performance deficiencies, as discussed above. Therefore, what is also needed in an approach to providing the heartbeat function in the context of a symmetric multiprocessor system.

SUMMARY OF THE INVENTION

A goal of the invention is to simultaneously satisfy the above-discussed requirements of improving and expanding the tight cluster concept which, in the case of the prior art, are not satisfied.

One embodiment of the invention is based on an apparatus, comprising: a shared memory unit; a first processing node coupled to said shared memory unit; and a second processing node coupled to said shared memory unit,

wherein the shared memory unit includes a range of addresses that are duplicated. Another embodiment of the invention is based on a method, comprising duplicating a shared address range in a shared memory unit that is coupled to a plurality of processing nodes. Another embodiment of the invention is based on an electronic media, comprising: a computer program adapted to duplicate a shared address range in a shared memory unit that is coupled to a plurality of processing nodes. Another embodiment of the invention is based on a computer program comprising computer program means adapted to perform the step of duplicating a shared address range in a shared memory unit that is coupled to a plurality of processing nodes when said computer program is run on a computer. Another embodiment of the invention is based on a system, comprising a multiplicity of processors, each with some private memory and all sharing some portion of memory, interconnected and arranged such that memory accesses to a first set of address ranges will be to local, private memory whereas memory accesses to a second set of address ranges will be to shared memory, and arranged such that at least a portion of one special range of shared memory which is duplicated so that two such are both written by each WRITE to that particular location, for which each READ of the secondary (mirrored) section is discarded or precluded. Another embodiment of the invention is based on a system comprised of a multiplicity of processing nodes, at least two shared-memory nodes, (SMNs) and means at each processing node to connect each processing node to each of said multiple SMNs. Said system to include means for communicating all Load and Store software instructions to shared memory to all of the essentially-identical shared memory nodes. Said system to include means for assuring that if any of said SMNs fails, the communication means ceases use of that SMN and notifies the attached processing node that the identified SMN has failed. Said system to assure that if atomic operations are performed affecting the SMN's that all secondary SMN's are corrected to contain the same value that the primary SMN contains.

These, and other goals and embodiments of the invention will be better appreciated and understood when considered in conjunction with the following

description and the accompanying drawings. It should be understood, however, that the following description, while indicating preferred embodiments of the invention and numerous specific details thereof, is given by way of illustration and not of limitation. Many changes and modifications may be made within the scope of the invention without departing from the spirit thereof, and the invention includes all such modifications.

BRIEF DESCRIPTION OF THE DRAWINGS

A clear conception of the advantages and features constituting the invention, and of the components and operation of model systems provided with the invention, will become more readily apparent by referring to the exemplary, and therefore nonlimiting, embodiments illustrated in the drawings accompanying and forming a part of this specification, wherein like reference characters (if they occur in more than one view) designate the same parts. It should be noted that the features illustrated in the drawings are not necessarily drawn to scale.

FIG. 1 illustrates a block schematic view of a share-as-needed basic system, representing an embodiment of the invention.

FIG. 2 illustrates a block schematic view of a share-as-needed highly available system, representing an embodiment of the invention.

FIG. 3 illustrates a block schematic view of a compute node dual port PCI adapter, representing an embodiment of the invention.

DESCRIPTION OF PREFERRED EMBODIMENTS

The invention and the various features and advantageous details thereof are explained more fully with reference to the nonlimiting embodiments that are illustrated in the accompanying drawings and detailed in the following description of preferred embodiments. Descriptions of well known components and processing techniques are omitted so as not to unnecessarily obscure the invention in detail.

The teachings of U.S. Ser. No. 09/273,430 include a system which is a single entity; one large supercomputer. The invention is also applicable to a cluster of workstations, or even a network.

The invention is applicable to systems of the type of Pfister or the type of U.S. Ser. No. 09/273,430 in which each processing node has its own copy of an operating system. The invention is also applicable to other types of multiple processing node systems.

The context of the invention can include a tight cluster as described in U.S. Ser. No. 09/273,430. A tight cluster is defined as a cluster of workstations or an arrangement within a single, multiple-processor machine in which the processors are connected by a high-speed, low-latency interconnection, and in which some but not all memory is shared among the processors. Within the scope of a given processor, accesses to a first set of ranges of memory addresses will be to local, private memory but accesses to a second set of memory address ranges will be to shared memory. The significant advantage to a tight cluster in comparison to a message-passing cluster is that, assuming the environment has been appropriately established, the exchange of information involves a single STORE instruction by the sending processor and a subsequent single LOAD instruction by the receiving processor.

The establishment of the environment, taught by U.S. Ser. No. 09/273,430 and more fully by companion disclosures (U.S. Provisional Application Ser. No. 60/220,794, filed July 26, 2000; U.S. Provisional Application Ser. No. 60/220,748, filed July 26, 2000; WSGR 15245-711; WSGR 15245-713; WSGR 15245-715; WSGR 15245-716; WSGR 15245-717; WSGR 15245-718; WSGR 15245-719; and WSGR 15245-720, the entire contents of all which are hereby expressly incorporated herein by reference for all purposes) can be performed in such a way as to require relatively little system overhead, and to be done once for many, many information exchanges. Therefore, a comparison of 10,000 instructions for message-passing to a pair of instructions for tight-clustering, is valid.

The invention can include an environment as described in U.S. Ser. No. 09/273,430 in which the second set of address ranges, the set of shared address

ranges, includes at least one range which is duplicated. WRITES to this memory range are written to both memories, and READS are read from a first but are ECC checked and, in the event of ECC failure on a READ, are then subsequently read from the second memory. On such a failure, an operator warning is issued.

Within such a system, mirroring a portion of shared memory can provide high availability, by which is meant protection from both hardware and software failures. The control of the mirrored memory requires that the operating system extensions and the application subsystem be written to appropriately utilize the shared memory.

Each processor can be provided with a very high-speed, low-latency communication means to the shared-memory. The low-latency communication means can include a communication link based on traces, cables and/or optical fiber(s). The low-latency communication means can include hardware (e.g., a circuit), firmware (e.g., flash memory) and/or software (e.g., a program). The communications means can include on-chip traces and/or waveguides. Further, each of these communication means can be duplicated.

The invention can include arranging the shared memory in banks. The duplicated data can segregated. The banks can be provided with separate interfaces. Further, the banks can be provided with duplicated interfaces.

Each processor in the system can also be provided with a specific inter-connection to the shared memory, arranged such that there are two connections to a particular range of memory addresses, and two banks of shared memory responsive to said particular range of addresses, said range referred to as the mirrored portion of said shared memory.

The invention can include providing the duplicated shared memory ranges with separate power supplies. Similarly, the invention can include providing the duplicated interfaces with separate power supplies. In-turn, the power supplies can be backed-up. The invention can include eliminating all single points of failure in a shared-memory as-needed computing system.

The provision of high availability is facilitated by several hardware and software features, the identity and function of which are taught by this

invention. A first, and most fundamental, of these features are provided by the nature of the close cluster system: a multiplicity of processors, each running its own copy of the operating system, each from its own private memory, exchanging information via shared memory.

A second feature is the provision of a heartbeat function between the processors of the system. The semaphore range can be used as an aid in failure recovery when accompanied by "heartbeat" mechanisms. When capturing a semaphore, this invention teaches the concept that the owning processor write its identification to the semaphore location. When subsequent heartbeat mechanisms indicate that a processor has failed, the processor detecting the heartbeat failure will search and release semaphores owned by the failed processor.

This can be implemented if the various nodes are able to determine when any node, processing or shared-facility, fails. In the preferred embodiment, the hardware subsystem, transparently to software, continuously monitors each node and informs at least two nodes when a failure does occur. Normal processing is then suspended for a few milliseconds while the nodes determine which is the failing element and prepare to recover.

A third feature is the mirroring of shared memory. Since memory is expensive, only the most important portion(s) of shared memory can be mirrored to reduce cost. Of course, more or all of shared memory can be mirrored. For example, a semaphore region that is used to pass signals between processors can be duplicated in the mirrored-memory region.

A fourth feature is that the interconnection between each processor and the mirrored memory is via a separate switch section. The separate switch sections can be separately powered. These sections can also be separately powered from conventional memory.

A fifth feature is that the operating system elements that deal with shared memory have no single point of failure. A companion disclosure describes extensions that can clean up semaphores and signaling elements by a process running on a first processor on behalf of a second, failed processor. Similarly, in the preferred embodiment the other shared resources held by the

failed processor are recorded in mirrored memory, and a companion cleanup process can release these facilities.

A sixth feature is the provision of sufficient power-supply redundancy to protect from failure of the supply. In the preferred embodiment, the protection will be provided by separate supplies for each processor and for the memories and for the separate switch to the mirrored memory.

A seventh feature is that the application or subsystem be structured to have no single point of failure. In practice, this can be achieved by snapshot saving the operation at recurring points, and storing all state information necessary to restore the application or subsystem to that point. Any updates beyond the snapshot must be such that they can be discarded without loss, or must be separately journaled (either into mirrored memory or to other reliable storage means) for full recovery to be possible.

This can be termed a recovery process. This requirement must be met by an application or an application enabler subsystem running on the basic system. The method utilized is to determine which state information is critical to a recovery process, to store that state at specific points, and journal all asynchronous events after that point. Then when a failure occurs, the enabler subsystem restores the saved state, factors in the journaled events, then resumes processing from that point.

An eighth feature is at least one multitailed storage facility, each with connections to at least two of the multiple processors in the system and to a disk, said disk to be utilized for all journaled information required by the application for the restoration of operation in the event of a failure. The purpose of the multitailed storage facility (multiple connection) is to access the disk via a second processor should the first fail. Unless the storage facility provides its own redundancy, at least two, mirrored, are required.

Shared-memory systems of the type described in US Patent Application Number 09/273,430, a filing assigned to *Times N Systems, Inc.*, are quite amenable to high-availability design. In a system of this type, each processing node is provided with a full set of privately-owned facilities, including processor, memory, and I/O devices. Therefore, only the most exceptional of

failures by one node with affect other nodes in the system. Only those elements are shared in such a system which need to be shared. These include a portion of memory and mechanisms to assure coordination of processing for shared tasks.

The design is therefore such that the failure of any single processing node can be prevented from causing system failure. The shared elements include a memory and an "atomic memory" in which a Load to a particular location causes not only a Load of that location but also a Store to that location. These shared facilities can be prevented from causing system failure if they are duplexed, and if each Load or Store to a shared facility is passed to each node of the duplexed pair. One is designated the primary and the other, the secondary shared node. Then, if the primary one of the duplexed pair fails, the second one has all of the state information that was in the failing one and thus a switch-over to the backup node allows operation to continue.

Each processing node should be provided with connectivity to both of the shared-facility nodes. The data which is Load from the secondary shared-facility node must be discarded at the processing node which issues the Load operation. To assure that competing Load operations to the atomic facility (which may not arrive in the same order at the secondary node as at the primary node) must result in the same data being stored in the corresponding locations in the two shared-facility nodes. Therefore, the Loading processor must Store back to an atomic location the data which it acquired from the primary shared node. In this way, the two shared-facility nodes will have the same data at any time that the primary node may fail.

THE PREFERRED EMBODIMENT

In the preferred embodiment, the system consists of a shared-memory node which includes an atomic complex and a "doorbell" signaling mechanism by which the processing nodes signal to each other. The hardware subsystem consists of PCI adapters which contain significant intelligence in hardware, with a connection mechanism between each of these PCI adapters and a companion set of PCI adapters in the primary shared memory node.

Each processing node is provided with a second PCI adapter or with a second channel out of it's single PCI adapter . The second channel is provided

with a connection mechanism to the secondary shared-memory node. The hardware subsystem passes information between the various processing nodes and the shared-memory node. In addition, the hardware subsystem continually monitors the node to which it is attached and the link to the companion node, and the hardware adapters continuously pass this information to each other.

Similarly, the hardware subsystem is set to a state in which the hardware subsystem can differentiate which shared-memory node is the primary and which is the secondary. Writes are passed to both shared memory nodes. PCI READs are also passed to both, but PCI READ responses from the secondary shared-memory node are discarded.

In the case where two separate PCI adapters are used in the processing node, the second node operates in a "stealth" mode, copying information sent by software to the primary PCI adapter but otherwise remaining quiescent at its processing-node interface, unless it determines that the secondary node or the connection to the secondary node is not properly operational, at which point it signals the software at both ends with an interrupt and relies on software to notify an operator that the backup system has failed.

For either case (one dual-connection PCI adapter or two separate adapters) software at the processing node re-programs the hardware to fully-activate the secondary connection when the primary connection reports significant problems.

Figure 1 is a drawing of an over-all share-as-needed system, showing multiple processor nodes, a single shared-memory node, and individual connection means connecting the processing nodes to the shared-memory node. With reference to figure 1, element 101 shows the processing nodes in the system. There can be multiple such nodes, as figure 101 shows. Element 102 shows the shared-memory node for the system of figure 1, and element 103 shows the links from the various processing nodes to the single shared-memory node.

Figure 2 shows a drawing of a system showing multiple processor nodes, two shared-memory nodes, and connection means linking each processing node to both shared-memory nodes. With reference to figure 2,

element 201 shows the processing nodes in the system. There can be multiple such nodes, as figure 201 shows. Element 202 shows the primary shared-memory node for the system of figure 2, and element 203 is the secondary shared-memory node in that system. Element 204 shows the links from the various processing nodes to both the primary and the secondary shared-memory nodes.

Figure 3 shows a drawing of a PCI adapter at a processing node, showing multiple link interfaces to the multiple shared-memory nodes. With reference to figure 3, element 301 shows the PCI Bus interface logic, and element 302 shows the address translator which determines whether a PCI Read or Write Command is intended for shared memory. Element 303 is the data buffers used for passing data to and from the PCI interface, and element 304 is the various control registers required to manage the operation of a PCI adapter.

Elements 305 and 307 are the send-side interfaces to the primary and secondary shared-memory units respectively, and elements 306 and 308 are the corresponding receive-side interfaces to the shared-memory units.

Element 309 directs the PCI Read and Write Commands to elements 305 and 307. In addition, element 309 accepts the results of those commands from elements 306 and 308. During normal operation, element 309 performs three functions. First, for ordinary PCI Read commands, it accepts the result from the primary SMN (if received) and if received correctly, 309 discards the result from the secondary SMN. For ordinary PCI Write commands, 309 accepts the acknowledgements from both SMN's to be sure both are received correctly. For atomic PCI Read commands, element 309 accepts the result from the primary SMN (if received) and if received correctly, 309 then compares the data to that received from the primary SMN. If they differ, element 309 issues a atomic Store to the addressed atomic location within the secondary SMN to assure that the two SMN's remain coherent.

For all of the above operations, if the secondary SMN fails to respond or produces an illegal response, element 309 notifies software, via one of the control registers, that the secondary SMN has failed, and then abandons operations to the secondary SMN. Similarly, for all of the above operations, if

the primary SMN fails to respond or produces an illegal response, element 309 notifies software, via one of the control registers, that the secondary SMN has failed, and then abandons operations to the primary SMN, and elevates the interface to the secondary SMN to primary status.

When notified by software to switch to the secondary SMN, the adapter of figure 3, through element 309, abandons operations to the primary SMN and begins operations to the secondary SMN.

This process could be done using two different adapters in each processing node. Three SMNs could be used in conjunction with majority logic at the processing node to detect additional failure modes. The primary SMN can provide the result of an atomic Read to the secondary SMN.

While not being limited to any particular performance indicator or diagnostic identifier, preferred embodiments of the invention can be identified one at a time by testing for the substantially highest performance. The test for the substantially highest performance can be carried out without undue experimentation by the use of a simple and conventional benchmark (speed) experiment.

The term substantially, as used herein, is defined as at least approaching a given state (e.g., preferably within 10% of, more preferably within 1% of, and most preferably within 0.1% of). The term coupled, as used herein, is defined as connected, although not necessarily directly, and not necessarily mechanically. The term means, as used herein, is defined as hardware, firmware and/or software for achieving a result. The term program or phrase computer program, as used herein, is defined as a sequence of instructions designed for execution on a computer system. A program may include a subroutine, a function, a procedure, an object method, an object implementation, an executable application, an applet, a servlet, a source code, an object code, and/or other sequence of instructions designed for execution on a computer system.

Practical Applications of the Invention

A practical application of the invention that has value within the technological arts is waveform transformation. Further, the invention is useful in conjunction with data input and transformation (such as are used for the

purpose of speech recognition), or in conjunction with transforming the appearance of a display (such as are used for the purpose of video games), or the like. There are virtually innumerable uses for the invention, all of which need not be detailed here.

Advantages of the Invention

A system, representing an embodiment of the invention, can be cost effective and advantageous for at least the following reasons. The invention improves the availability of parallel computing systems. The invention imporves the reliability of parallel computing systems. The invention also improves the scalability of parallel computing systems.

All the disclosed embodiments of the invention described herein can be realized and practiced without undue experimentation. Although the best mode of carrying out the invention contemplated by the inventors is disclosed above, practice of the invention is not limited thereto. Accordingly, it will be appreciated by those skilled in the art that the invention may be practiced otherwise than as specifically described herein.

For example, although the high-availability, shared-memory cluster described herein can be a separate module, it will be manifest that the high-availability, shared-memory cluster may be integrated into the system with which it is associated. Furthermore, all the disclosed elements and features of each disclosed embodiment can be combined with, or substituted for, the disclosed elements and features of every other disclosed embodiment except where such elements or features are mutually exclusive.

It will be manifest that various additions, modifications and rearrangements of the features of the invention may be made without deviating from the spirit and scope of the underlying inventive concept. It is intended that the scope of the invention as defined by the appended claims and their equivalents cover all such additions, modifications, and rearrangements.

The appended claims are not to be interpreted as including means-plus-function limitations, unless such a limitation is explicitly recited in a given claim using the phrase "means for." Expedient embodiments of the invention are differentiated by the appended subclaims.

CLAIMS

What is claimed is:

1. An apparatus, comprising:

a shared memory unit;

a first processing node coupled to said shared memory unit; and

a second processing node coupled to said shared memory unit,

wherein the shared memory unit includes a range of addresses that are duplicated.

2. The apparatus of claim 1, wherein said shared memory units includes a first bank and a second bank, data is said second bank duplicating data in said first bank.

3. The apparatus of claim 1, further comprising a first duplicated interface coupled between said first processing node and said shared memory unit and a second duplicated interface coupled between said first processing node and said shared memory unit.

4. The apparatus of claim 1, further comprising a duplicate power supply coupled to said shared memory unit.

5. The apparatus of claim 1, further comprising a multitailed storage facility coupled to said first processing node and said second processing node; and a computer-readable medium coupled to said multitailed storage facility.

6. A computer system comprising the apparatus of claim 1.

7. A method, comprising duplicating a shared address range in a shared memory unit that is coupled to a plurality of processing nodes.

8. The method of claim 7, wherein WRITES to said shared address range are written to a first memory range and a second memory range, and READS are read from said first memory range.

9. The method of claim 8, wherein READS are read from said first address range after error correction code checking.

10. The method of claim 9, wherein, in an event of error correction code failure on a READ, said READ is read from the second address range.

11. The method of claim 10, wherein, in said event an operator warning is issued.

12. An electronic media, comprising: a computer program adapted to duplicate a shared address range in a shared memory unit that is coupled to a plurality of processing nodes.

13. A computer program comprising computer program means adapted to perform the step of duplicating a shared address range in a shared memory unit that is coupled to a plurality of processing nodes when said computer program is run on a computer.

14. A computer program as claimed in claim 13, embodied on a computer-readable medium.

15. A system, comprising a multiplicity of processors, each with some private memory and all sharing some portion of memory, interconnected and arranged such that memory accesses to a first set of address ranges will be to local, private memory whereas memory accesses to a second set of address ranges will be to shared memory, and arranged such that at least a portion of one special range of shared memory which is duplicated so that two such are

both written by each WRITE to that particular location, for which each READ of the secondary (mirrored) section is discarded or precluded.

16. The system of claim 15, wherein a heartbeat function is provided between the processors using the shared memory.

17. The system of claim 16, wherein a semaphore region is included in the mirrored memory range.

18. The system of claim 17, wherein the switch which connects the shared memory to the processing units is duplexed.

19. The system of claim 16, wherein a processor can clean up semaphores left by a companion processor which has stopped its heartbeat.

20. The system of claim 15, further comprising redundant mass storage.

21. A system comprised of a multiplicity of processing nodes, at least two shared-memory nodes, (SMNs) and means at each processing node to connect each processing node to each of said multiple SMNs. Said system to include means for communicating all Load and Store software instructions to shared memory to all of the essentially-identical shared memory nodes. Said system to include means for assuring that if any of said SMNs fails, the communication means ceases use of that SMN and notifies the attached processing node that the identified SMN has failed. Said system to assure that if atomic operations are performed affecting the SMN's that all secondary SMN's are corrected to contain the same value that the primary SMN contains.

22. The system of claim 21, wherein the means for maintaining correctness are in PCI adapters in the processing nodes.

23. The system of claim 21, wherein the means for assuring that atomic operations remain coherent across the multiple SMNs is performed via SMN-SMN transfer means rather than from the processing node means.

24. The system of claim 21, wherein a sequence number is associated with each transfer from a processing node to the SMNs and wherein failure is defined to include failure to respond after a defined period of time, and also is defined to include mismatch of returned sequence number.

25. The system of claim 21, wherein the atomic operations to all except the primary SMN are delayed and are translated at the processing node to be non-atomic update operations, which then are transferred to the alternative SMNs to assure that the atomic values remain coherent.

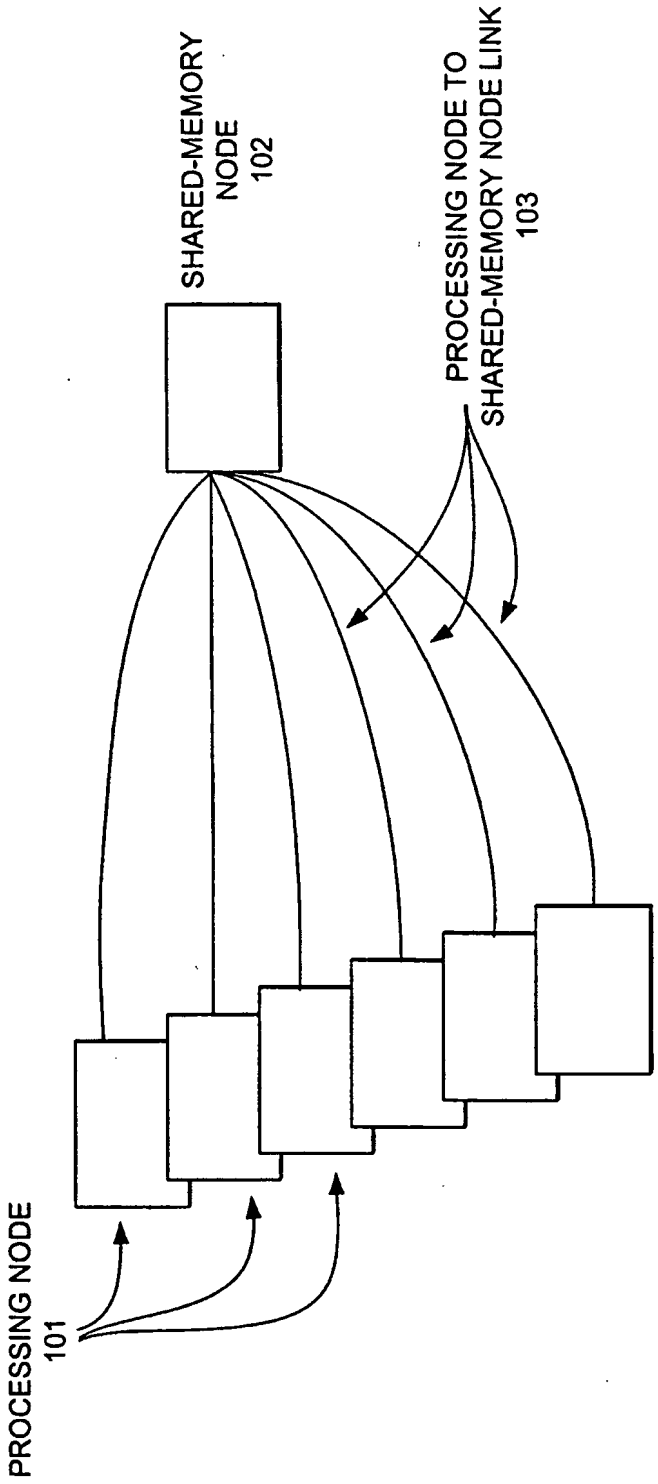
SHARED-MEMORY NODE 102
PROCESSING NODE TO SHARED-MEMORY NODE LINK 103
PROCESSING NODE 101

FIG. 1

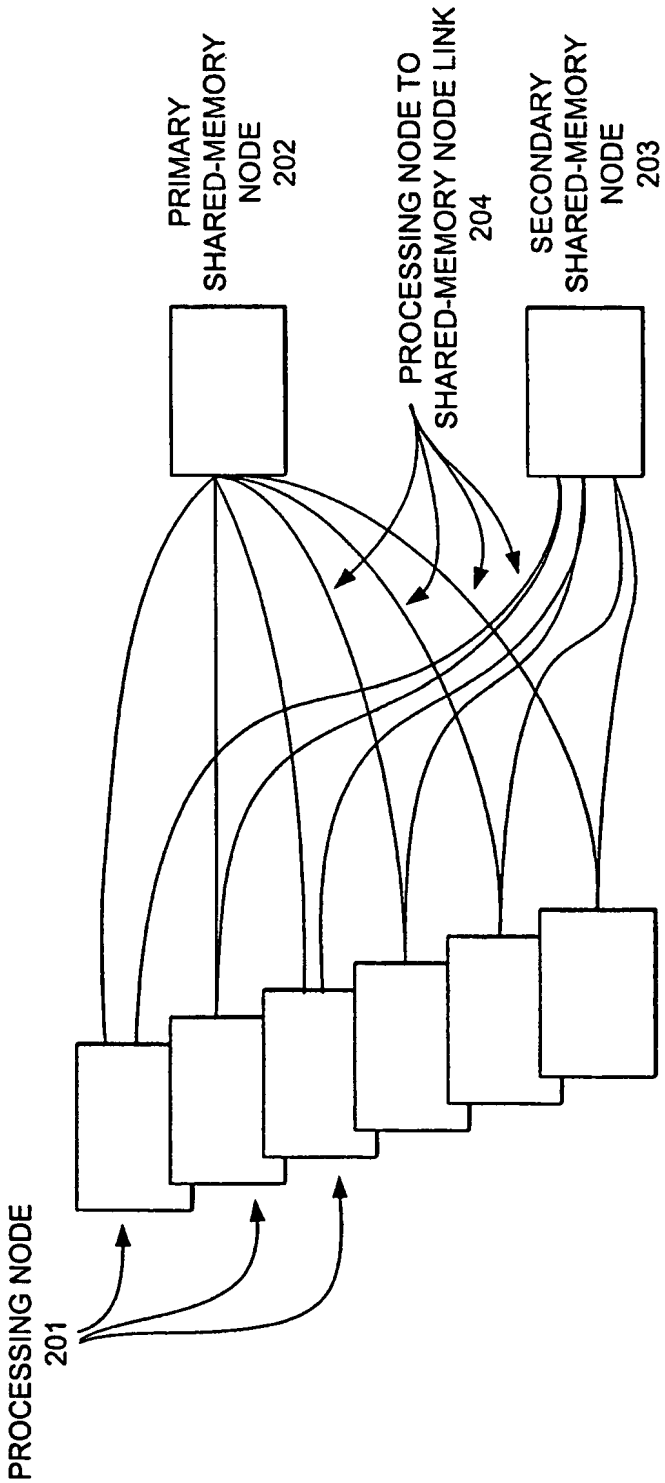
PRIMARY SHARED-MEMORY NODE 202
PROCESSING NODE TO SHARED-MEMORY NODE LINK 204
SECONDARY SHARED-MEMORY NODE 203
PROCESSING NODE 201

FIG. 2

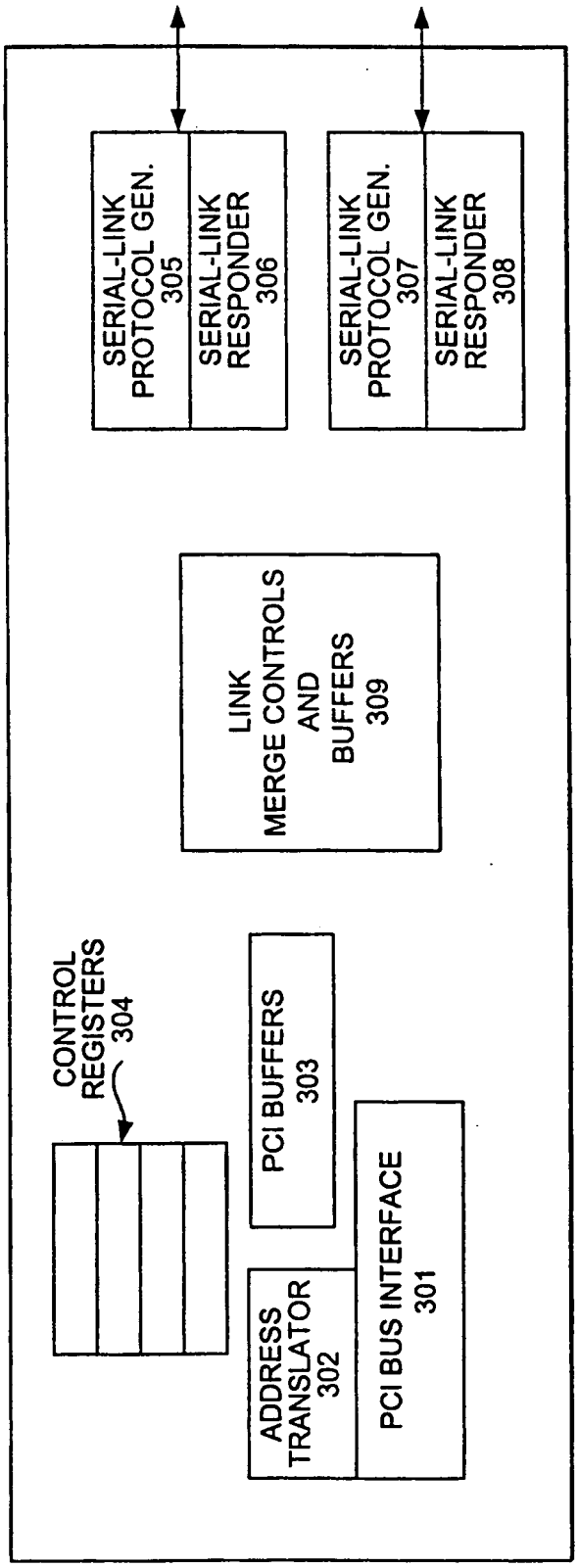
SERIAL-LINK PROTOCOL GEN. 305
SERIAL-LINK RESPONDER 306
SERIAL-LINK PROTOCOL GEN. 307
SERIAL-LINK RESPONDER 308
LINK MERGE CONTROLS AND BUFFERS 309
CONTROL REGISTERS 304
PCI BUFFERS 303
ADDRESS TRANSLATOR 302
PCI BUS INTERFACE 301

FIG. 3